Terms used **branch dummy null NOP** Found **139** of **171,143**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 139 Result page: **1** 2 3 4 5 6 7 next

Relevance scale

1 Extending Java for high-level Web service construction

Aske Simon Christensen, Anders Møller, Michael I. Schwartzbach
November 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 6
Publisher: ACM Press
Full text available: pdf(947.02 KB) Additional Information: full citation, abstract, references, citings, index terms

We incorporate innovations from the <bigwig> project into the Java language to provide high-level features for Web service programming. The resulting language, JWIG, contains an advanced session model and a flexible mechanism for dynamic construction of XML documents, in particular XHTML. To support program development we provide a suite of program analyses that at compile time verify for a given program that no runtime errors can occur while building documents or receiving form input, and ...

Keywords: Interactive Web services, XML, data-flow analysis

2 Exploiting multi-way branching to boost superscalar processor performance

Yen-Jen Oyang
January 1991 **ACM SIGPLAN Notices**, Volume 26 Issue 3
Publisher: ACM Press
Full text available: pdf(504.97 KB) Additional Information: full citation, abstract, index terms

This paper discusses exploiting multi-way branching to boost superscalar processor performance. The exploitation of multi-way branching contributes to a boost in superscalar processor performance through two effects: (1) increase of instruction-level parallelism and (2) reducing of the amount of branch penalty. The work presented in this paper comprises two conjunctive parts. The first part is a compiler technique called the SV (Shadow Variable) transformation. The second part is a new multi-way ...

3 Operational semantics-directed compilers and machine architectures

John Hannan
July 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 16 Issue 4
Publisher: ACM Press
Full text available: pdf(2.13 MB) Additional Information: full citation, abstract, references, index terms

We consider the task of automatically constructing intermediate-level machine

architectures and compilers generating code for these architectures, given operational semantics for source languages. We use operational semantics in the form of abstract machines given by rewrite systems in which the rewrite rules operate on terms representing states of computations. To construct compilers and new architectures we employ a particular strategy called pass separation, a form of staging transformat ...

Keywords: abstract machines, pass separation, semantics-based compilation

4 Efficient instruction scheduling using finite state automata

Vasanth Bala, Norman Rubin

December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture**

Publisher: IEEE Computer Society Press

Full text available: pdf(1.34 MB) Additional Information: full citation, references, citings, index terms

5 Active memory: a new abstraction for memory system simulation

Alvin R. Lebeck, David A. Wood

January 1997 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 7 Issue 1

Publisher: ACM Press

Full text available: pdf(690.38 KB) Additional Information: full citation, references, citings, index terms

Keywords: Cache memory, direct-execution simulation, memory hierarchy, on-the-fly simulation, trace-driven simulation

6 SCR algorithm: saving/restoring states of file systems

Wei Xiao-Hui, Ju Jiu-Bin

January 1999 **ACM SIGOPS Operating Systems Review**, Volume 33 Issue 1

Publisher: ACM Press

Full text available: pdf(555.58 KB) Additional Information: full citation, abstract, index terms

Fault-tolerance is very important in cluster computing. Many famous cluster-computing systems have implemented fault-tolerance by using checkpoint/restart mechanism. But existent checkpointing algorithms can not restore the states of a file system when roll-backing the running of a program, so there are many restrictions on file accesses in existent fault-tolerance systems. SCR algorithm, an algorithm based on atomic operation and consistent schedule, which can restore the states of file systems ...

Keywords: atomic operation, checkpointing, fault-tolerance, recoverability of file systems

7 Register assignment through resource classification for ASIP microcode generation

Clifford Liem, Trevor May, Pierre Paulin

November 1994 **Proceedings of the 1994 IEEE/ACM international conference on Computer-aided design**

Publisher: IEEE Computer Society Press

Full text available: pdf(653.28 KB) Additional Information: full citation, abstract, references, citings, index terms

Application Specific Instruction-Set Processors (ASIPs) offer designers the ability for high-speed data and control processing with the added flexibility needed for late design

specifications, accommodation of design errors, and product evolution. However, code generation for ASIPs is a complex problem and new techniques are needed for its success. The register assignment task can be a critical phase, since often in ASIPs, the number and functionality of available registers is limited, as t ...

8 Optimization for a superscalar out-of-order machine

Anne M. Holler

December 1996 **Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture**

Publisher: IEEE Computer Society

Full text available: pdf(1.55 MB) Additional Information: full citation, abstract, citings, index terms

Compiler optimization plays a key role in unlocking the performance of the PA-8000, an innovative dynamically-scheduled machine which is the first implementation of the 64-bit PA 2.0 member of the HP PA-RISC architecture family. This wide superscalar, long out-of-order machine provides significant execution bandwidth and automatically hides latency at runtime; however, despite its ample hardware resources, many of the optimizing transformations which proved effective for the PA-8000 served to au ...

9 A structured approach for the definition of the semantics of active databases

Piero Fraternali, Letizia Tanca

December 1995 **ACM Transactions on Database Systems (TODS)**, Volume 20 Issue 4

Publisher: ACM Press

Full text available: pdf(4.15 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Active DBMSs couple database technology with rule-based programming to achieve the capability of reaction to database (and possibly external) stimuli, called events. The reactive capabilities of active databases are useful for a wide spectrum of applications, including security, view materialization, integrity checking and enforcement, or heterogeneous database integration, which makes this technology very promising for the near future. An active database system consists of ...

Keywords: active database systems, database rule processing, events, fixpoint semantics, rules, semantics

10 Performance evaluation and prediction for parallel algorithms on the BBN GP1000

François Bodin, Daniel Windheiser, William Jalby, Daya Atapattu, Mannho Lee, Dennis Gannon

June 1990 **ACM SIGARCH Computer Architecture News , Proceedings of the 4th international conference on Supercomputing ICS '90**, Volume 18 Issue 3b

Publisher: ACM Press

Full text available: pdf(1.17 MB) Additional Information: full citation, abstract, references, citings, index terms

The techniques of "load/store" memory reference modeling is based on deriving performance characteristics of the memory architecture of a computer by looking at the behavior of simple sequences of load, store and nop (null operation) instructions. The resulting data base can be used to match load/store templates against algorithm kernels to predict performance or as a source of data for testing analytical models of the architecture. In this paper we study the BBN GP1000 parallel ...

11 Safety checking of machine code

Zhichen Xu, Barton P. Miller, Thomas Reps

May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation PLDI '00**, Volume 35

Issue 5
Publisher: ACM Press

Full text available: pdf(306.71 KB) Additional Information: full citation, abstract, references, citings, index terms

We show how to determine statically whether it is safe for untrusted machine code to be loaded into a trusted host system. Our safety-checking technique operates directly on the untrusted machine-code program, requiring only that the initial inputs to the untrusted program be annotated with typestate information and linear constraints. This approach opens up the possibility of being able to certify code produced by any compiler from any source language, which gives the code prod ...

12 Virtual machine monitors: Implementing an untrusted operating system on trusted hardware

David Lie, Chandramohan A. Thekkath, Mark Horowitz
October 2003 **Proceedings of the nineteenth ACM symposium on Operating systems principles**
Publisher: ACM Press

Full text available: pdf(280.87 KB) Additional Information: full citation, abstract, references, citings, index terms

Recently, there has been considerable interest in providing "trusted computing platforms" using hardware~---~TCPA and Palladium being the most publicly visible examples. In this paper we discuss our experience with building such a platform using a traditional time-sharing operating system executing on XOM~---~a processor architecture that provides copy protection and tamper-resistance functions. In XOM, only the processor is trusted; main memory and the operating system are not trusted.Our opera ...

Keywords: XOM, XOMOS, untrusted operating systems

13 Sentinel PRE: Hoisting beyond Exception Dependency with Dynamic Deoptimization

Rei Odaira, Kei Hiraki
March 2005 **Proceedings of the international symposium on Code generation and optimization CGO '05**
Publisher: IEEE Computer Society

Full text available: pdf(205.02 KB) Additional Information: full citation, abstract

Many excepting instructions cannot be removed by existing Partial Redundancy Elimination (PRE) algorithms because the ordering constraints must be preserved between the excepting instructions, which we call exception dependencies. In this work, we propose Sentinel PRE, a PRE algorithm that overcomes exception dependencies and retains program semantics. Sentinel PRE first hoists excepting instructions without considering exception dependencies, and then detects exception reordering by fast analys ...

14 Operational transformation: Grouping in collaborative graphical editors

Claudia-Lavinia Ignat, Moira C. Norrie
November 2004 **Proceedings of the 2004 ACM conference on Computer supported cooperative work**
Publisher: ACM Press

Full text available: pdf(212.15 KB) Additional Information: full citation, abstract, references, index terms

Often collaborative graphical systems lag behind well accepted single-user applications in terms of features supported. The frequently used operations of group/ungroup offered by almost every single-user graphical editor have not been considered by the collaborative graphical editing systems that try to preserve the intentions of the users involved in the concurrent editing. In this paper we present a novel algorithm based on operation serialisation for consistency maintenance in collaborativ ...

Keywords: collaborative graphical editors, consistency, grouping/ungrouping, maintenance, serialisation

15 Efficient instruction scheduling for delayed-load architectures

Steven M. Kurlander, Todd A. Proebsting, Charles N. Fischer
September 1995 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 17 Issue 5
Publisher: ACM Press

Full text available: pdf(2.30 MB) Additional Information: full citation, abstract, references, citings, index terms, review

A fast, optimal code-scheduling algorithm for processors with a delayed load of one instruction cycle is described. The algorithm minimizes both execution time and register use and runs in time proportional to the size of the expression-tree. An extension that spills registers when too few registers are available is also presented. The algorithm also performs very well for delayed loads of greater than one instruction cycle. A heuristic that schedules DAGs and is based on our optimal expres ...

16 Measuring the dynamic behaviour of AspectJ programs

Bruno Dufour, Christopher Goard, Laurie Hendren, Oege de Moor, Ganesh Sittampalam, Clark Verbrugge
October 2004 **ACM SIGPLAN Notices , Proceedings of the 19th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '04**, Volume 39 Issue 10
Publisher: ACM Press

Full text available: pdf(226.86 KB) Additional Information: full citation, abstract, references, index terms

This paper proposes and implements a rigorous method for studying the dynamic behaviour of AspectJ programs. As part of this methodology several new metrics specific to AspectJ programs are proposed and tools for collecting the relevant metrics are presented. The major tools consist of: (1) a modified version of the AspectJ compiler that tags bytecode instructions with an indication of the cause of their generation, such as a particular feature of AspectJ; and (2) a modified version of the *J ...

Keywords: AspectJ, aspect-oriented programming, dynamic metrics, java, optimization, performance, program analysis

17 Effectiveness of a machine-level, global optimizer

Mark S. Johnson, Terrence C. Miller
July 1986 **ACM SIGPLAN Notices , Proceedings of the 1986 SIGPLAN symposium on Compiler construction SIGPLAN '86**, Volume 21 Issue 7
Publisher: ACM Press

Full text available: pdf(853.18 KB) Additional Information: full citation, abstract, references, citings, index terms

We present an overview of the design of a machine-code-level, global (intraprocedural) optimizer that supports several front-ends producing code for the Hewlett-Packard Precision Architecture family of machines. The basic optimization strategy is described, including information about the division of responsibilities between various components of the compiler. Optimization algorithms are described, including a discussion of the dataflow information they require. Measurements showing the col ...

18 Workshop on architectural support for security and anti-virus (WASSA): Towards the issues in architectural support for protection of software execution

Weidong Shi, Hsien-Hsin S. Lee, Chenghuai Lu, Mrinmoy Ghosh
March 2005 **ACM SIGARCH Computer Architecture News**, Volume 33 Issue 1
Publisher: ACM Press
Full text available: pdf(436.30 KB) Additional Information: full citation, abstract, references, index terms

Recently, there is a growing interest in the research community to employ tamper-resistant processors for software protection. Many of these proposed systems rely on a specially tailored secure processor to prevent 1) illegal software duplication, 2) unauthorized software modification, and 3) unauthorized software reverse engineering. Most of these works primarily focus on the feasibility demonstration and design details rather than trying to elucidate many fundamental issues that are either "el ...

Keywords: attack, copy protection, encryption, security, tamper resistance

19 Intrusion detection and prevention: On deriving unknown vulnerabilities from zero-day polymorphic and metamorphic worm exploits
Jedidiah R. Crandall, Zhendong Su, S. Felix Wu
November 2005 **Proceedings of the 12th ACM conference on Computer and communications security CCS '05**
Publisher: ACM Press
Full text available: pdf(334.95 KB) Additional Information: full citation, abstract, references, index terms

Vulnerabilities that allow worms to hijack the control flow of each host that they spread to are typically discovered months before the worm outbreak, but are also typically discovered by third party researchers. A determined attacker could discover vulnerabilities as easily and create zero-day worms for vulnerabilities unknown to network defenses. It is important for an analysis tool to be able to generalize from a new exploit observed and derive protection for the vulnerability.Many researcher ...

Keywords: honeypots, metamorphism, polymorphic worms, polymorphism, symbolic execution, worms

20 Lambda, the ultimate label or a simple optimizing compiler for Scheme
William D. Clinger, Lars Thomas Hansen
July 1994 **ACM SIGPLAN Lisp Pointers , Proceedings of the 1994 ACM conference on LISP and functional programming LFP '94**, Volume VII Issue 3
Publisher: ACM Press
Full text available: pdf(1.20 MB) Additional Information: full citation, abstract, references, citings, index terms

Optimizing compilers for higher-order languages need not be terribly complex. The problems created by non-local, non-global variables can be eliminated by allocating all such variables in the heap. Lambda lifting makes this practical by eliminating all non-local variables except for those that would have to be allocated in the heap anyway. The eliminated non-local variables become local variables that can be allocated in registers. Since calls to known procedures are just gotos that pass ar ...

Results 1 - 20 of 139 Result page: **1** 2 3 4 5 6 7 next

Terms used **instruction profiling dummy null NOP**

Found **11** of **9 searched out of 43,400.**

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 11 of 11

Relevance scale

1 A Hardware-Software Platform for Intrusion Prevention

Milenko Drinic, Darko Kirovski

December 2004 **Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture MICRO 37**

Publisher: IEEE Computer Society

Full text available: pdf(254.63 KB) Additional Information: full citation, abstract

Preventing execution of unauthorized software on a given computer plays a pivotal role in system security. The key problem is that although a program at the beginning of its execution can be verified as authentic, its execution flow can be redirected to externally injected malicious code using, for example, a buffer overflow exploit. We introduce a novel, simplified, hardware-assisted intrusion prevention platform. Our platform introduces overlapping of program execution and MAC verification. It ...

2 System-level power optimization: techniques and tools

Luca Benini, Giovanni de Micheli

April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 5 Issue 2

Publisher: ACM Press

Full text available: pdf(385.22 KB) Additional Information: full citation, abstract, references, citings, index terms

This tutorial surveys design methods for energy-efficient system-level design. We consider electronic sytems consisting of a hardware platform and software layers. We consider the three major constituents of hardware that consume energy, namely computation, communication, and storage units, and we review methods of reducing their energy consumption. We also study models for analyzing the energy cost of software, and methods for energy-efficient software design and compilation. This survery ...

3 An Accurate Instruction-Level Energy Consumption Model for Embedded RISC Processors

Sheayun Lee, Andreas Ermedahl, Sang Lyul Min

August 2001 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN workshop on Languages, compilers and tools for embedded systems LCTES '01 , Proceedings of the 2001 ACM SIGPLAN workshop on Optimization of middleware and distributed systems OM '01**, Volume 36 Issue 8

Publisher: ACM Press

Full text available: pdf(224.01 KB) Additional Information: full citation, abstract, references, citings, index

terms

Energy consumption of software is becoming an increasingly important issue in designing mobile embedded systems where batteries are used as the main power source. As a consequence, recently, a number of promising techniques have been proposed to optimize software for reduced energy consumption. Such low-power software techniques require an energy consumption model that can be used to estimate or predict the energy consumed by software. We propose a technique to derive an accurate energy consu ...

4 Efficient instruction scheduling using finite state automata

Vasanth Bala, Norman Rubin

December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture**

Publisher: IEEE Computer Society Press

Full text available: pdf(1.34 MB) Additional Information: full citation, references, citings, index terms

5 Lambda, the ultimate label or a simple optimizing compiler for Scheme

William D. Clinger, Lars Thomas Hansen

July 1994 **ACM SIGPLAN Lisp Pointers , Proceedings of the 1994 ACM conference on LISP and functional programming LFP '94**, Volume VII Issue 3

Publisher: ACM Press

Full text available: pdf(1.20 MB) Additional Information: full citation, abstract, references, citings, index terms

Optimizing compilers for higher-order languages need not be terribly complex. The problems created by non-local, non-global variables can be eliminated by allocating all such variables in the heap. Lambda lifting makes this practical by eliminating all non-local variables except for those that would have to be allocated in the heap anyway. The eliminated non-local variables become local variables that can be allocated in registers. Since calls to known procedures are just gotos that pass ar ...

6 Supporting autonomic computing functionality via dynamic operating system kernel aspects

Michael Engel, Bernd Freisleben

March 2005 **Proceedings of the 4th international conference on Aspect-oriented software development AOSD '05**

Publisher: ACM Press

Full text available: pdf(381.44 KB) Additional Information: full citation, abstract, references, index terms

To master the complexity of software systems in the presence of unexpected events potentially affecting system operation, the *Autonomic Computing Initiative* [16] aims to build systems that have the ability to control and organize themselves to meet unforeseen changes in the hard- and software environment.The basic principles employed by autonomic computing are self-configuration, self-optimization, self-healing and self-protection. Typically, these principles are cross-cutting concerns, s ...

Keywords: NetBSD, autonomic computing, dynamic aspects, operating system kernel, organic computing

7 VDL—a Definition system for all levels

John A. N. Lee

December 1973 **ACM SIGARCH Computer Architecture News , Proceedings of the 1st annual symposium on Computer architecture ISCA '73**, Volume 2 Issue 4

Publisher: ACM Press

Full text available: pdf(666.75 KB) Additional Information: full citation, abstract, references, citings, index terms

The VDL system for the description of programming languages which was originally used for the definition of PL/I is extended to the description of processors. This paper shows the relationship between the language of definition and the abstract machine over which the semantics of the language are specified. It is demonstrated that the level of description can be chosen to suit the various needs of the computing community, each level being well nested within its outer level, whilst using onl ...

8 Extending Java for high-level Web service construction

Aske Simon Christensen, Anders Møller, Michael I. Schwartzbach
November 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 6
Publisher: ACM Press

Full text available: pdf(947.02 KB) Additional Information: full citation, abstract, references, citings, index terms

We incorporate innovations from the <bigwig> project into the Java language to provide high-level features for Web service programming. The resulting language, JWIG, contains an advanced session model and a flexible mechanism for dynamic construction of XML documents, in particular XHTML. To support program development we provide a suite of program analyses that at compile time verify for a given program that no runtime errors can occur while building documents or receiving form input, and ...

Keywords: Interactive Web services, XML, data-flow analysis

9 A Structural View of PL/I

David Beech
March 1970 **ACM Computing Surveys (CSUR)**, Volume 2 Issue 1
Publisher: ACM Press
Full text available: pdf(2.86 MB) Additional Information: full citation, references, citings, index terms

10 Session 4: WORM vs. WORM: preliminary study of an active counter-attack mechanism

Frank Castaneda, Emre Can Sezer, Jun Xu
October 2004 **Proceedings of the 2004 ACM workshop on Rapid malcode**
Publisher: ACM Press
Full text available: pdf(289.95 KB) Additional Information: full citation, abstract, references, index terms

Self-propagating computer worms have been terrorizing the Internet for the last several years. With the increasing density, inter-connectivity and bandwidth of the Internet combined with security measures that inadequately scale, worms will continue to plague the Internet community. Existing anti-virus and intrusion detection systems are clearly inadequate to defend against many recent fast-spreading worms. In this paper we explore an active counter-attack method - anti-worms. We propose a me ...

Keywords: anti-worm, good worm, worm

11 An iteratively structured information processor

Gerald R. Kane
December 1974 **ACM SIGARCH Computer Architecture News , Proceedings of the 2nd annual symposium on Computer architecture ISCA '75**, Volume 3 Issue 4

Publisher: ACM Press

Full text available: pdf(636.96 KB) Additional Information: full citation, abstract, references

There exists a widely held belief that the full potential of LSI technology will be realized by arrays of logic circuits. A novel machine architecture is proposed that consists of a uniform array of identical cells with the property that the array executes a high-level programming language directly. Some of the more important features of such machines are their inherent abilities to sustain concurrent processes and to maintain efficient information storage. A detailed simulation of a language ma ...

Results 1 - 11 of 11

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	2	"5941985".pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 14:38
S2	2	"5964867".pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 15:59
S3	1	2002/0010913	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 16:00
S4	2	"20020010913"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 16:03
S5	194	(insert$3 add$3) with (dummy null) with instruction	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:14
S6	135	S5 and @ad < "20011128"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 16:15
S7	0	S6 and skid	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 16:11
S8	839	ambiguity with creat$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:13

EAST Search History

S9	0	S6 and S8	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 16:12
S10	1	S5 and S8	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 16:13
S11	9	S6 and profil$3	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/16 16:15
S12	289	(insert$3 add$3 pad$4) with (dummy null) with instruction$1	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:12
S13	165	ambiguity with creat$4 and (profil$3)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:23
S14	28755	(insert$3 add$3) with instruction	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:14
S15	5	S13 and S14	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:14
S16	1	S12 and S13	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:23

EAST Search History

S17	30585	((ambiguity with creat$4) branch) and (profil$3)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:24
S18	7	S17 and S12	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2006/03/17 08:24